Министерство образования и молодежной политики Свердловской области



ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт по программе «**Практическое занятие 10**»

Выполнил: Максимов Кирилл Эдуардович

Группа: ПР-21

Преподаватель: Мирошниченко Г.В

2023

**Задание:**

**Входные и выходные данные**

**Листинг программы (если есть)**

**Скрин-шот экранов**

**Тестовые ситуации**

**Вывод**

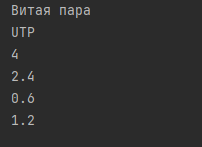
abstract class AbstractCable {  
 abstract var cable: String  
 abstract var type: String  
 abstract var countCore: Int  
 abstract var D: Double  
 abstract var Q: Double  
 abstract fun GetInfo()  
  
 constructor(\_cable: String, \_type: String, \_countCore: Int, \_D: Double){  
 cable = \_cable  
 type = \_type  
 countCore = \_countCore  
 D = \_D  
 }  
  
 constructor(\_cable: String, \_type: String){  
 cable = \_cable  
 type = \_type  
 }  
}

class HeirCable (cable: String, type: String, countCore: Int, D: Double, var P: Boolean) : AbstractCable(cable, type, countCore, D)  
{  
 override var cable: String = cable  
 override var type: String = type  
 override var countCore: Int = countCore  
 override var D: Double = D  
 override var Q = D / countCore.toDouble()  
  
 override fun GetInfo() {  
 *println*(cable)  
 *println*(type)  
 *println*(countCore)  
 *println*(D)  
 *println*(Q)  
 }  
 fun getQp(): Double {  
 return if(P) 2.0 \* Q else 0.7 \* Q  
 }  
  
}

abstract class AbstractOnInterface: InterfaceCable {  
 override var cable = "Витая пара"  
 override var type = "UTP"  
 override var countCore = 4  
 override var D = 2.4  
 override var Q = D / countCore.toDouble()  
  
 override fun GetInfo() {  
 *println*(cable)  
 *println*(type)  
 *println*(countCore)  
 *println*(D)  
 *println*(Q)  
 }  
}

fun main() {  
 var Qp: HeirCable = HeirCable("Витая пара", "UTP", 4, 2.4, true)  
  
 Qp.GetInfo()  
 *println*(Qp.getQp())  
}

interface InterfaceCable {  
 var cable: String  
 var type: String  
 var countCore: Int  
 var D: Double  
 var Q: Double  
 fun GetInfo()  
}



Изучил абстрактные классы в Kotlin.